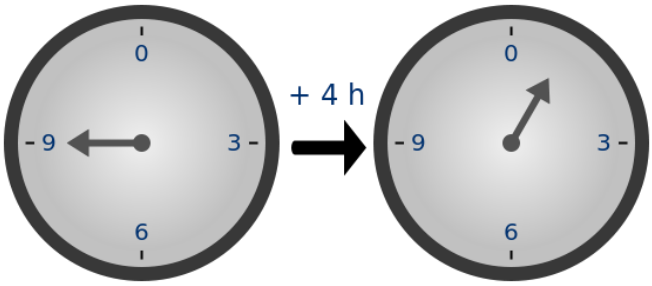
Aritmética Modular

En matemática, la aritmética modular es un sistema aritmético para clases de equivalencia de números enteros llamadas clases de congruencia. La aritmética modular fue introducida en 1801 por Carl Friedrich Gauss.
Algunas veces se le llama, sugerentemente, aritmética del reloj, ya que los números «dan la vuelta» tras alcanzar cierto valor llamado módulo.



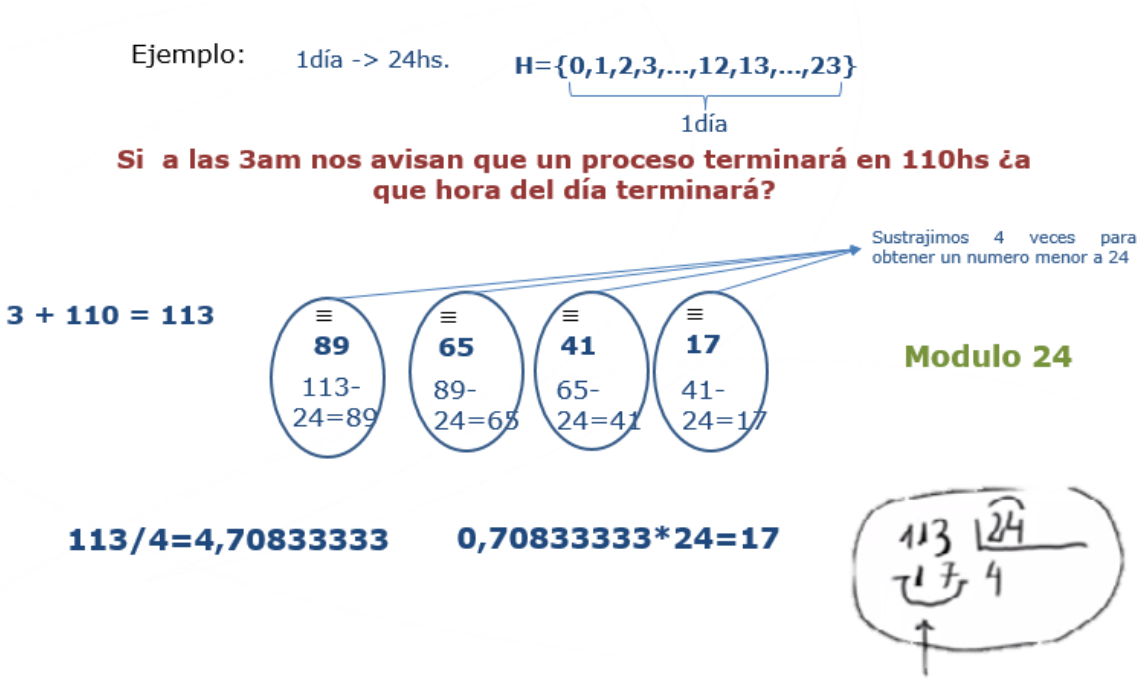
La Aritmética Modular es una de las aplicaciones más importantes de la teoría de números. Y está representada por la función mod.
La función módulo representa el resto de la división. Por ejemplo a mod b significa que queremos hallar el resto de a, que representamos como: a mod b = a – b {a/b}

La Aritmética Modular también define un sistema de numeración.
Veamos por ejemplo la secuencia: 0 mod 3, 1 mod 3, 2 mod 3, 3 mod 3, 4 mod 3....
Evaluando tenemos 0, 2, 1, 0, 1... que equivale a la numeración en base 3.

Algunas de las propiedades más importantes de la Aritmética Modular.
Suma, resta, multiplicacion, Division (no esta definida para todos los caos)

Existen muchas aplicaciones de la Aritmética Modular, por ejemplo en el calendario los días de la semana corresponden a una aritmética módulo 7, las horas, minutos y segundos corresponden al módulo 60.
Hallar el último dígito de un número decimal corresponde a una aritmética módulo 10.

Ejemplo de cómo ayudaría



Cuestiones Prácticas

Está representada por la función mod (% en Java).
La función módulo representa el resto de la división y queda definida por $a \bmod b = a - b\{a/b\}$.

Propiedades:

Suma: **(x + y) mod m = (x mod m + y mod m) mod m**.

Resta: La resta es sólo la suma con valores negativos por los que **(x – y) mod m = (x mod m – y mod m) mod m**.

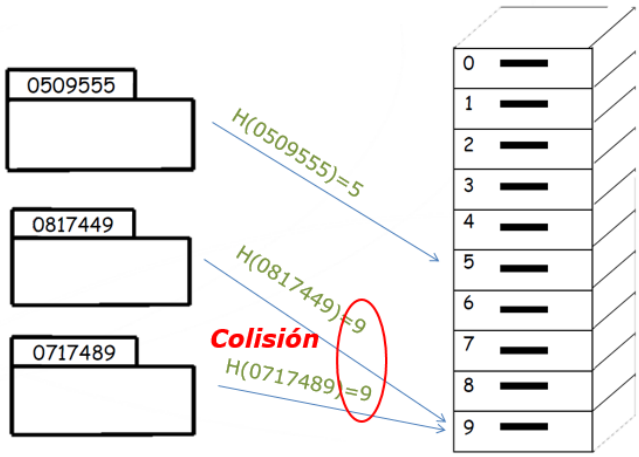
Multiplicación: La multiplicación **xy mod m = (x mod m)(y mod m) mod m**.
Esto se da debido a que la multiplicación es simplemente una suma repetida.

División: No existe el inverso de la multiplicación genérico. *(pero existen algoritmos que permiten su cálculo en los casos que si existe)*

Congruencia o Equivalencia: $a \equiv b(mod\ n)$ Se lee como que **a** es equivalente a **b modulo n**.

$$a_1 \equiv b_1(mod\ n)$$
$$a_2 \equiv b_2(mod\ n)$$
$$a_1 + a_2 \equiv b_1 + b_2(mod\ n)$$
$$a_1 * a_2 \equiv b_1 * b_2(mod\ n)$$

Algunas de sus aplicaciones son en Tabla Hash y en criptografía.
Por ejemplo Dado un código k, para conocer el sitio donde se almacena, se utiliza la función: **h(k)= k mod 10** .



Exponenciación modular

El operador es el equivalente aritmético a calcular una exponenciación natural y luego calcular el modulo o residuo del resultado respecto a cierto número (M).
La Exponenciación Modular recibe tres operandos de entrada:
• El operando X es llamado comúnmente la base de la exponenciación,
• El numero Y es referido como el exponente
• M es referido como el modulo de la representación.

Definición matemática:

Sean X, Y y M números enteros tal que: $M > 0$.
Se define la Exponenciación Modular P como:

$P \equiv X^Y\ mod\ M$ si y solo si existe un número entero k tal que:

$X^Y = k.M + P$

Propiedad recursiva:

$$\mathbf{x} = a^b mod\ n = (a\ mod\ n)a^{b-1} mod\ n$$

Ejemplo:

Calculo normal: $4^5 mod\ 7 = 1024\ mod\ 7 = 2$

Calculo recursivo: $4\ mod\ 7 = 4$; $(4 * 4)\ mod\ 7 = 2$; $(2 * 4)\ mod\ 7 = 1$; $(1 * 4)\ mod\ 7 = 4$; $(4 * 4)\ mod\ 7 = 2$ *(implementado 5 veces)*

La Exponenciación Modular es un operador de amplio uso en técnicas criptográficas tales como:
• RSA • Rabin • Elgamal • McEliece
Básicamente, el operador es el equivalente aritmético a calcular una exponenciación natural y luego calcular el modulo o residuo del resultado respecto a cierto número (M).

La Exponenciación Modular debe operar sobre números enteros de gran tamaño, generalmente en binario, esto es debido a que por niveles de seguridad cada vez más exigentes en los sistemas criptográficos modernos, obliga a un aumento paulatino en el tamaño en bits de las claves, de modo que los operadores que componen el sistema deben manejar operandos de mayor tamaño.
Esta situación tiende a hacer lenta la ejecución de algunos de estos operadores, especialmente en aquellos de gran complejidad.
Existe entonces un interés por la optimización de tales operaciones, de modo que se puedan conseguir tiempos de ejecución aceptables.
Las implementaciones en hardware aparecen como alternativa de diseño, ya que el hardware dedicado y optimizado para una operación es potencialmente más rápido que la ejecución de una aplicación de software sobre hardware genérico.

La idea detrás de la exponenciación modular rápida consiste en obtener la representación del exponente n en dígitos binarios (dt,dt-1,...,d2,d1), con dt =1, y hallar los distintos cuadrados sucesivos (mod m) de la base a: (a1,a2,a4,...,a2t), para después multiplicar módulo m las potencias a2i correspondientes a los dígitos binarios di que sean "1".

Pequeño Teorema de Fermat

Pierre Fermat fue un abogado que vivió en Francia de 1601 a 1665. Se interesó por las matemáticas cuando ya estaba en los treintas y siempre se acercó a ella como pasatiempo; sin embargo, los pasatiempos de los genios son superiores a toda una vida de dedicación de gentes menos dotadas. Trabajó mucho con Teoría de los Números, una rama de las matemáticas que podemos identificar con la Aritmética, en particular con los relojes de Gauss, y realizó algunos descubrimientos importantes.

> Si p es un número primo, entonces, para cada número natural a, con $a>0$, $a^p \equiv a \pmod p$
>
> Pierre de Fermat, 1636

Aunque son equivalentes, el teorema suele ser presentado de esta otra forma:

> Si p es un número primo, entonces, para cada número natural a, con $a>0$, coprimo con p , $a^{p-1} \equiv 1 \pmod p$
>
> Pierre de Fermat, 1636

Esto quiere decir que, si se eleva un número **a** a la **p-ésima** potencia y al resultado se le resta **a**, lo que queda es divisible por **p** *(aritmética modular)*. Su interés principal está en su aplicación al problema de la primalidad y en criptografía.

Este teorema no tiene nada que ver con el legendario último teorema de Fermat, que fue sólo una conjetura durante 350 años y finalmente fue demostrado por Andrew Wiles en 1995.

MCD (Máximo Común divisor)

En matemáticas, se define el máximo común divisor *(abreviado mcd o "M.C.D")* de dos o más números enteros al mayor número que los divide sin dejar resto.

Hay dos formas de hallar el mcd, tenemos una forma larga , que sería buscando todos los divisores de los números y lo mismo en el caso de los múltiplos, hasta encontrar el común. Pero de manera mucho mas rápida se puede resolver mediante la división por números primos.

Esta es la forma larga *(o para cuando aún no se conocen los números primos)*, de buscar el **mcd**. Buscamos los divisores de cada número, y seleccionamos entre todos los divisores comunes *(que se repiten en los dos grupos)* el mayor de ellos será el **mcd**.

Div (12)____{1, 2, 3, 4, 6, 12}

Div (18)____{1, 2, 3, 6, 9, 18}

Una de las formas cortas de llegar al mismo resultado es hallando el **mcd** de ambos números para ello tomamos solamente los que son comunes en los dos, en este ejemplo hemos seleccionado solo un 2 porque solo uno es común y un 3 por la misma razón, el 5 no lo utilizamos porque no es común en los dos, y realizamos la multiplicación (2.3=6).

m.c.d (12,18)= 6

12	2
6	2
3	3
1	

12= 2.2 . 3

18	2
9	3
3	3
1	

18= 2. 3. 3

m.c.d (12,18)= 2 . 3= 6

Descomponemos en factores primos los dos números y tomamos los factores comunes a ambos con el menor exponente con el que aparezcan.
El problema inicial es el siguiente:
Encontrar el máximo común divisor entre dos números enteros positivos a y b.

Aunque es un método bastante útil y sencillo para conseguir lo que queremos tiene un evidente problema: si los números son muy grandes, o si sus factores primos lo son, se complica ya que el cálculo de la descomposición se torna bastante tedioso.

MCD – Algoritmo de Euclides

El **algoritmo de Euclides** nos servirá para esta tarea:

- Tomando **a** y **b**, dividimos el más grande por el mas pequeño *(suponemos a el mas grande y b el más pequeño)*; y nos proporcionará un cociente, **c1**, y un resto, **r1**.
- Si r1=0 , entonces mcd(a,b)=b. *(asunto resuelto)*
- Si no es cero dividimos el divisor, c1, entre el resto, r1, obteniendo otro cociente,c2, y otro resto,r2.
- Si r2 = 0, entonces mcd(a.b) = r1 .
- Si no es cero volvemos a dividir divisor entre resto.

Y así sucesivamente.

Esto es, el máximo común divisor entre a y b es el último resto distinto de cero que obtengamos con el procedimiento anterior.
Si analizamos el algoritmo de Euclides se ve claramente que necesitamos demostrar que el máximo común divisor entre a y b es igual al máximo común divisor entre b y r 1 .
De este modo, esa igualdad se mantendrá durante todo el proceso y llegaremos a que el último resto distinto de cero es el máximo común divisor de los dos enteros positivos iniciales.

Teorema:

- El máximo común divisor de dos números enteros positivos a y b, con a > b > 0, coincide con el máximo común divisor de b y r, siendo r el resto que se obtiene al dividir a entre b.

Demostración:

Sean $d = mcd(a, b)$ y $t = mcd(b, r)$

Vamos a demostrar que $d = t$.

Por definición de máximo común divisor, se tiene que d es un divisor tanto de a como de b. Por tanto $a = a_1 d$ y $b = b_1 d$

Por otro lado, por el algoritmo de la división se tiene que:

$a = bq + r$, con $r < b$ (1)

Por tanto d es un divisor de r. Como ya teníamos que también es un divisor de b entonces debe dividir a su máximo común divisor, esto es, d **es un divisor de** t.

Por otro lado, t es un divisor tanto de b como de r. Por ello se tiene que $b = pt$ y r=st. Sustituyendo estas dos igualdades en (1) obtenemos lo siguiente:

$$a = ptq + st = (pq + s)t$$

Por tanto t es un divisor de a. Como también lo era de b debe ser un divisor de su máximo común divisor, es decir, t es un divisor de d.

Como d es un divisor de t y t es un divisor de d no queda otra opción más que $\boldsymbol{t = d}$.

Como ya teníamos que también es un divisor de b entonces debe dividir a su máximo común divisor, esto es, d es un divisor de t.
Por otro lado, t es un divisor tanto de b como de r.
Por ello se tiene que b = pt y r = st .
Sustituyendo estas dos igualdades en (1) obtenemos lo siguiente:

a = ptq + st = (pq + s)t

Por tanto t es un divisor de a. Como también lo era de b debe ser un divisor de su máximo común divisor, es decir, t es un divisor de d.
Como d es un divisor de t y t es un divisor de d no queda otra opción más que t = d.
Por tanto el algoritmo de Euclides funciona.

MCD – Ejemplo

Cálculo de mcd(721, 448)

Como hemos explicado antes dividimos el número mayor entre el menor; si el resto no es cero dividimos el divisor entre el resto; y así sucesivamente hasta que llegamos a un punto en el que el resto es cero. Los resultados de las divisiones (expresados como dividendo=divisor • cociente + resto) son:

		div	mod
721	448	1	273
448	273	1	175
273	175	1	98
175	98	1	77
98	77	1	21
77	21	3	14
21	14	1	7 *
14	7	2	0

Como marca el *, se tiene que mcd(721, 448) = 7, el último divisor que no es nulo.

Un ejemplo de un algoritmo que realice esta función puede ser:

```
funcion mcd-euclides (a,b:entero):entero;
        div,resto,restoant:entero;
        resto=1;
        {
        mientras (resto != 0) hacer
                        restoant=resto;
                        resto = a mod b;
                        a=b;
                        b=resto;
                fin mientras ;
        retornar restoant;
        }
```

	a	b	div	resto	restoant
1	721	448	1	273	1
2	448	273	1	175	273
3	273	175	1	98	175
4	175	98	1	77	98
5	98	77	1	21	77
6	77	21	3	14	21
7	21	14	1	7	14
8	14	7	2	0	**7**

Inverso Multiplicativo

El inverso multiplicativo, recíproco e inversa de un número ***x*** no nulo, es el número, denotado como $1/x$ o x^{-1}, que multiplicado por ***x*** da ***1*** como resultado.
El multiplicador modular inverso de un entero ***n*** módulo ***p*** es un entero ***m***, tal que:

$$n^{-1} \equiv m(mod\ p)$$

Esto significa que es el inverso multiplicativo en el anillo de los enteros módulo ***p***. Es equivalente a

$$m.n \equiv 1(mod\ p)$$

El multiplicador inverso de ***n*** módulo ***p*** existe si y sólo si ***n*** y ***p*** son **coprimos**, es decir, si MCD(n, p)=1.

> En matemáticas, dos números enteros a y b son números primos entre sí (o **coprimos**, o primos relativos), si no tienen ningún factor primo en común, o, dicho de otra manera, si no tienen otro divisor común más que 1 y -1. Equivalentemente son primos entre sí, si y sólo si, su máximo común divisor es igual a 1.

Si existe el multiplicador modular inverso de ***n*** módulo ***p***, se puede definir la operación de división entre ***n*** módulo ***p*** mediante la multiplicación por el inverso.

El 0 no tiene inverso multiplicativo. Todo número complejo, salvo el 0, tiene un inverso que es un número complejo. El inverso de un número real también es real, y el de un número racional también es racional.
El inverso multiplicativo es aplicable a distintos tipos de objetos matemáticos.

- La inversa de una matriz es otra matriz, denominada matriz inversa, que al multiplicarse por la original es igual a la matriz identidad.
- La inversa de una función es la resultante de despejar la variable independiente, convirtiéndola en dependiente. Gráficamente es un trazado paralelo a la recta diagonal y = x.
- En las Matemáticas Constructivas, para que un número real x tenga inverso, no es suficiente que sea falso que x = 0. Además, debe existir un número racional r tal que 0 < r < |x|.
- En Aritmética Modular, el inverso multiplicativo de x también está definido: es el número a tal que (a × x) mod n = 1. ***Sin embargo, este inverso multiplicativo sólo existe si a y n son primos entre sí***.

Por ejemplo, el inverso de 3 módulo 11 es 4, porque es la solución de (3 × x) mod 11 = 1. Un algoritmo empleado para el cálculo de inversos modulares es el Algoritmo extendido de Euclides.

Números Primos

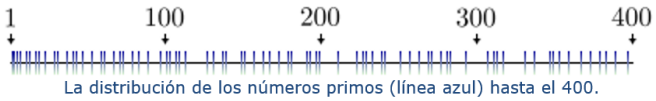
La razón más importante es la que está expresada en el Teorema fundamental de la Aritmética:

> **Todo número natural N puede expresarse como producto de números primos, N = p1p2…pn, y esta representación es única salvo por el orden.**

Por ejemplo, *4 = 2×2, 12 = 2x2x3, 500 = 100×5 = 2x50x5 = 2x5x10x5 = 2x5x2x5x2 = 2x2x2x5x5. Las dos últimas expresiones son la misma, salvo el orden en que están presentados los factores primos 2 y 5*.
Otro número para factorizar podría ser **1001 = 7×143 = 7x11x13**. Con este número batallamos un poquito. ¿Y con 6901? ¿Y con 280123?
Tomando en cuenta las dificultades para poder factorizar un número, podemos pensar que el teorema será todo lo fundamental que quieran los matemáticos, pero es de poca utilidad.

Ahora pensemos, si es tan difícil factorizar números, ¿entonces de qué nos sirve? Si alguien estuvo pensando previamente en esta objeción, acaba de descubrir el secreto de la criptografía.

Los números primos están metidos en la lista de los números naturales de una manera azarosa. Las claves no son no obvias, para que los vayamos descubriendo. Si hacemos una lista de los que nuestra paciencia nos permite, escribiremos algo así como 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, etc.



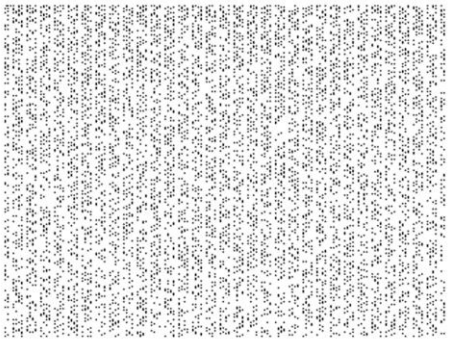
La distribución de los números primos (línea azul) hasta el 400.

¿Cuántos hay? ¿Es una lista infinita la de los números primos?

Un teorema tan viejo como la cultura griega nos dice que los **números primos son infinitos**. La prueba es por reducción al absurdo, es decir, se supone que la lista es finita, y de ahí se argumenta para llegar a una contradicción. El argumento es sencillo: supongamos que tenemos un número finito de primos p1, p2, p3,…pN. Consideremos ahora el número M = (el producto de todos los primos) + 1 = (p1 p2 p3…pN) + 1, y cuestionemos si M es primo o si no lo es. Es claro que M es mayor que cualquiera de los números primos, puesto que el producto de dos números mayores que 1 es mayor que cualquiera de los factores. Si M fuera primo, entonces tendríamos un número primo mayor que todos los números primos, en particular mayor que sí mismo, y eso es una contradicción.

Si M no es primo, entonces tenemos una cantidad infinita de números especiales, los primos, distribuidos de una manera aparentemente aleatoria en la lista de todos los números naturales.

La **dificultad para factorizar un número cualquiera en sus factores primos** es precisamente la **ventaja** que utilizan los desarrolladores de algoritmos para codificar los números de tarjetas de crédito, para almacenar bases de datos encriptadas, etc.


La distribución de todos los números primos comprendidos entre 1 y 76.800, de izquierda a derecha y de arriba abajo. Cada pixel representa un número. Los píxeles negros representan números primos; los blancos representan números no primos.